Great topic 👌 — this is **WSTG-SESS-05: Testing for Session Fixation**.

**🧪 Testing for Session Fixation**

**🔎 What is Session Fixation?**

Session Fixation happens when an attacker **sets or predicts a victim’s session ID** before login.

* If the app **doesn’t regenerate session IDs after login**, the attacker can use the fixed ID to hijack the victim’s authenticated session.

Example flow:

1. Attacker gets a valid session ID (from login page, URL, cookie).
2. Sends a link to victim with that **predefined session ID**.
3. Victim logs in → server **accepts the same session ID**.
4. Attacker now uses that session ID to impersonate victim.

**🔧 How to Test**

**1. Check Session ID before and after login**

* Intercept login request with **Burp Suite**.
* Compare:
  + Session token from login page (unauthenticated).
  + Session token after authentication (authenticated).
* 🔴 If **same ID persists across login → vulnerable**.
* 🟢 If **new ID generated after login → safe**.

**2. Test Token Delivery Mechanisms**

* See if app accepts session tokens from:
  + **URL parameters** → https://site.com/login?session=12345
  + **Hidden form fields** → <input type="hidden" name="sessionid">
  + **Cookies** → Set-Cookie: SESSION=12345

All of these can be manipulated by attacker.

**3. Try Fixing Your Own Token**

* Use Burp’s **Repeater** to send requests with a manually chosen session ID.
* If the server **accepts attacker-supplied session IDs** → Fixation possible.

**4. Look for Weak Session Management Practices**

* Long-lived session tokens.
* Session not invalidated on logout.
* Multiple concurrent sessions allowed with the same token.

**🚨 Exploitation Example (Lab-safe)**

1. Attacker logs in anonymously → receives JSESSIONID=attacker123.
2. Attacker tricks victim to click:
3. https://vulnerable.com/login;jsessionid=attacker123
4. Victim logs in → session bound to attacker’s ID.
5. Attacker reuses attacker123 to access victim’s account.

**✅ Mitigations (Best Practices)**

* **Regenerate session ID after login and privilege change.**
* Accept session tokens **only in cookies**, not URL/hidden fields.
* Use **Secure, HttpOnly, SameSite** cookie flags.
* Expire and invalidate old sessions on logout.
* Detect abnormal session reuse (multiple IPs/agents).

⚡ **Summary:**  
Session Fixation = attacker forces victim to authenticate using a session ID **controlled by attacker**.  
Test by comparing session IDs before & after login in Burp Suite.

👉 Do you want me to also give you a **Burp Suite step-by-step lab workflow** for detecting session fixation (like a checklist you can repeat in practice)?